

03 037
YOR

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日

Date of Application:

2003年 3月20日

出 願 番 号

Application Number:

特願2003-079010

[ST.10/C]:

[JP 2003-079010]

出 願 人

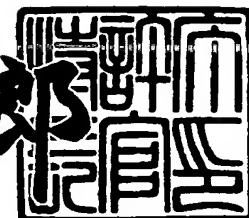
Applicant(s):

インターナショナル・ビジネス・マシーンズ・コーポレーション

2003年 5月20日

特 許 庁 長 官
Commissioner,
Japan Patent Office

太田 信一郎



出証番号 出証特2003-3037042

【書類名】 特許願

【整理番号】 JP9030037

【提出日】 平成15年 3月20日

【あて先】 特許庁長官 殿

【国際特許分類】 G06F 9/45

【発明者】

【住所又は居所】 神奈川県大和市下鶴間 1 6 2 3 番地 1 4 日本アイ・ビー・エム株式会社 東京基礎研究所内

【氏名】 小笠原 武史

【特許出願人】

【識別番号】 390009531

【氏名又は名称】 インターナショナル・ビジネス・マシーンズ・コーポレーション

【代理人】

【識別番号】 100086243

【弁理士】

【氏名又は名称】 坂口 博

【代理人】

【識別番号】 100091568

【弁理士】

【氏名又は名称】 市位 嘉宏

【代理人】

【識別番号】 100108501

【弁理士】

【氏名又は名称】 上野 剛史

【復代理人】

【識別番号】 100104156

【弁理士】

【氏名又は名称】 龍華 明裕

【手数料の表示】

【予納台帳番号】 053394

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【包括委任状番号】 9706050

【包括委任状番号】 9704733

【包括委任状番号】 0207860

【ブルーフの要否】 要

【書類名】 明細書

【発明の名称】 コンパイラ装置、コンパイラプログラム、記録媒体、及びコンパイル方法

【特許請求の範囲】

【請求項 1】 コンパイル対象のプログラムである対象プログラムの例外処理を最適化するコンパイラ装置であって、

前記対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハンドラ検出部と、

検出された前記複数捕捉例外ハンドラが捕捉する前記複数の例外のうち、前記複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する例外選択部と、

前記例外選択部により選択された例外の組を前記複数捕捉例外ハンドラに代わって捕捉し、前記共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成部と

を備えるコンパイラ装置。

【請求項 2】 前記例外ハンドラ生成部は、前記代替例外ハンドラにおいて、前記共通の処理に移行させる分岐命令を生成し、生成した前記分岐命令によって前記共通の処理に移行させる請求項 1 記載のコンパイラ装置。

【請求項 3】 前記例外選択部は、前記複数捕捉例外ハンドラにおいて発生する頻度が予め定められた基準頻度以上であり、かつ前記共通の処理に移行する例外の組を選択する請求項 1 記載のコンパイラ装置。

【請求項 4】 前記例外選択部は、前記複数捕捉例外ハンドラにおいて例外が発生する頻度として、前記複数捕捉例外ハンドラが実行される回数当たりの、前記複数捕捉例外ハンドラにおいて前記例外の組の何れかが発生する回数を検出する請求項 3 記載のコンパイラ装置。

【請求項 5】 前記共通の処理は、関数呼び出しにより呼び出された関数の処理中に発生した例外を前記関数呼び出しの外で捕捉する処理を含み、

前記複数例外捕捉ハンドラは、前記関数呼び出しの内部で例外を捕捉し、

前記例外選択部は、前記共通の処理から前記複数捕捉例外ハンドラに至る関数呼び出しの入れ子の深さが、予め定められた数より大きいことを更に条件として、前記例外の組を選択する請求項 1 記載のコンパイラ装置。

【請求項 6】 前記例外選択部は、前記複数捕捉例外ハンドラから前記共通の処理に移行する間に処理が経由する他の例外ハンドラの数、予め定められた数より大きいことを更に条件として、前記例外の組を選択する請求項 1 記載のコンパイラ装置。

【請求項 7】 前記共通の処理は、関数呼び出しにより呼び出された関数の処理中に発生した例外を前記関数呼び出しの外で捕捉する処理を含み、

前記複数例外捕捉ハンドラは、前記関数呼び出しの内部で例外を捕捉し、

前記例外選択部は、前記共通の処理から前記複数捕捉例外ハンドラに至る関数呼び出しの入れ子の深さと、前記複数捕捉例外ハンドラから前記共通の処理に移行する間に処理が経由する他の例外ハンドラの数とに更に基づき、前記例外の組を選択する請求項 1 記載のコンパイラ装置。

【請求項 8】 前記例外ハンドラ検出部は、一の前記複数捕捉例外ハンドラと、前記一の複数捕捉例外ハンドラにおいて発生した少なくとも一の例外を捕捉する他の前記複数捕捉例外ハンドラとの 2 つの前記複数捕捉例外ハンドラを検出し、

前記例外選択部は、前記一の複数捕捉例外ハンドラが捕捉する複数の例外のうち、前記他の複数捕捉例外ハンドラにおいて捕捉した例外を再び発生させることにより前記共通の処理に移行する例外の組を選択し、

前記例外ハンドラ生成部は、前記 2 つの複数捕捉例外ハンドラのそれぞれに対応付けて、2 つの前記代替例外ハンドラのそれぞれを生成し、前記 2 つの複数捕捉例外ハンドラのそれぞれに代えて、対応する前記 2 つの代替例外ハンドラのそれぞれにおいて、前記例外選択部により選択された例外の組を捕捉させる請求項 1 記載のコンパイラ装置。

【請求項 9】 コンパイル対象のプログラムである対象プログラムの例外処理を最適化するコンパイラ装置としてコンピュータを機能させるコンパイラプログラムであって、

前記コンピュータを、

前記対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハンドラ検出部と、

検出された前記複数捕捉例外ハンドラが捕捉する前記複数の例外のうち、前記複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する例外選択部と、

前記例外選択部により選択された例外の組を前記複数捕捉例外ハンドラに代わって捕捉し、前記共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成部と

して機能させるコンパイラプログラム。

【請求項 1 0】 前記共通の処理は、関数呼び出しにより呼び出された関数の処理中に発生した例外を前記関数呼び出しの外で捕捉する処理を含み、

前記複数例外捕捉ハンドラは、前記関数呼び出しの内部で例外を捕捉し、

前記例外選択部は、前記共通の処理から前記複数捕捉例外ハンドラに至る関数呼び出しの入れ子の深さと、前記複数捕捉例外ハンドラから前記共通の処理に移行する間に処理が経由する他の例外ハンドラの数とに更に基づき、前記例外の組を選択する請求項 9 記載のコンパイラプログラム。

【請求項 1 1】 前記例外ハンドラ検出部は、一の前記複数捕捉例外ハンドラと、前記一の複数捕捉例外ハンドラにおいて発生した少なくとも一の例外を捕捉する他の前記複数捕捉例外ハンドラとの 2 つの前記複数捕捉例外ハンドラを検出し、

前記例外選択部は、前記一の複数捕捉例外ハンドラが捕捉する複数の例外のうち、前記他の複数捕捉例外ハンドラにおいて捕捉した例外を再び発生させることにより前記共通の処理に移行する例外の組を選択し、

前記例外ハンドラ生成部は、前記 2 つの複数捕捉例外ハンドラのそれぞれに対応付けて、2 つの前記代替例外ハンドラのそれぞれを生成し、前記 2 つの複数捕捉例外ハンドラのそれぞれに代えて、対応する前記 2 つの代替例外ハンドラのそれぞれにおいて、前記例外選択部により選択された例外の組を捕捉させる請求項

9 記載のコンパイラプログラム。

【請求項 1 2】 請求項 9 から請求項 1 1 の何れかに記載のコンパイラプログラムを記録した記録媒体。

【請求項 1 3】 コンパイル対象のプログラムである対象プログラムの例外処理を最適化するコンパイラ装置としてコンピュータを動作させるコンパイル方法であって、

前記対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハンドラ検出段階と、

検出された前記複数捕捉例外ハンドラが捕捉する前記複数の例外のうち、前記複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する例外選択段階と、

前記例外選択部により選択された例外の組を前記複数捕捉例外ハンドラに代わって捕捉し、前記共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成段階と

を備えるコンパイル方法。

【発明の詳細な説明】

【0 0 0 1】

【発明の属する技術分野】

本発明は、コンパイラ装置、コンパイラプログラム、記録媒体、及びコンパイル方法に関する。特に本発明は、例外処理を最適化するコンパイラ装置、コンパイラプログラム、記録媒体、及びコンパイル方法に関する。

【0 0 0 2】

【従来の技術】

近年、プログラムの保守性及び堅牢性を高める目的で、例外処理を記述することのできるプログラム言語が普及してきている。このようなプログラム言語で記述されたプログラムは、プログラムにおいて例外が発生した場合に、プログラム中の例外が発生した箇所から、当該例外に対応する例外ハンドラに処理を移す。

従来、例外ハンドラに処理を移すこの処理を最適化する技術として、頻繁に発

生する例外の種類を検出し、検出した種類の例外を発生する命令を、例外ハンドラへの分岐命令に書き換える技術が用いられている（非特許文献1参照。）。

【0003】

【非特許文献1】

小笠原 武史、小松 秀昭、及び中谷 登志男著” A Study of Exception Handling and Its Dynamic Optimization in Java（登録商標）” Object-Oriented Programming Systems, Languages, and Applications (OOPSLA2001) 会議録 2001年、邦題「Java（登録商標）における、例外処理及び例外処理の動的最適化の研究」

【0004】

【発明が解決しようとする課題】

しかしながら、上記技術は、発生させる例外の種類を特定できない場合に、例外ハンドラに処理を移す処理を最適化することができない。例えば、複数の種類の例外を捕捉する例外ハンドラが、捕捉した例外を再び発生させる場合には、コンパイラ装置は、何れの種類の例外を捕捉するかを予め定めることができないので、例外を再び発生させる命令をどのように最適化すべきかを決定することができない。

そこで本発明は、上記の課題を解決することのできるコンパイラ装置、コンパイラプログラム、記録媒体、及びコンパイル方法を提供することを目的とする。この目的は特許請求の範囲における独立項に記載の特徴の組み合わせにより達成される。また従属項は本発明の更なる有利な具体例を規定する。

【0005】

【課題を解決するための手段】

即ち、本発明の第1の形態によると、コンパイル対象のプログラムである対象プログラムの例外処理を最適化するコンパイラ装置であって、対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハン

ドラ検出部と、検出された複数捕捉例外ハンドラが捕捉する複数の例外のうち、複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する例外選択部と、例外選択部により選択された例外の組を複数捕捉例外ハンドラに代わって捕捉し、共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成部とを備えるコンパイラ装置、コンパイラプログラム、記録媒体、及びコンパイル方法を提供する。

なお上記の発明の概要は、本発明の必要な特徴の全てを列挙したものではなく、これらの特徴群のサブコンビネーションも又発明となりうる。

【 0 0 0 6 】

【発明の実施の形態】

以下、発明の実施の形態を通じて本発明を説明するが、以下の実施形態は特許請求の範囲にかかる発明を限定するものではなく、又実施形態の中で説明されている特徴の組み合わせの全てが発明の解決手段に必須であるとは限らない。

図 1 は、コンパイラ装置 1 0 の機能ブロック図を示す。コンパイラ装置 1 0 は、コンパイル対象のプログラムである対象プログラムのコンパイル時において、例外処理を最適化する装置であり、例外ハンドラ検出部 1 0 0 と、例外選択部 1 1 0 と、例外ハンドラ生成部 1 2 0 とを備える。例外ハンドラ検出部 1 0 0 は、対象プログラムを取得すると、当該対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出し、検出結果を例外選択部 1 1 0 に送る。

【 0 0 0 7 】

そして、例外選択部 1 1 0 は、例外ハンドラ検出部 1 0 0 により検出された複数捕捉例外ハンドラが捕捉する複数の例外のうち、複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択し、選択結果を例外ハンドラ生成部 1 2 0 に送る。そして、例外ハンドラ生成部 1 2 0 は、例外選択部 1 1 0 により選択された例外の組を複数捕捉例外ハンドラに代わって捕捉して共通の処理に移行する代替例外ハンドラを生成する。更に、例外ハンドラ生成部 1 2 0 は、生成した当該代替例外ハンドラにおいて、共通の処理に移行さ

せる分岐命令を生成し、生成した分岐命令によって、共通の処理に移行させる。
 続いて、例外ハンドラ生成部 1 2 0 は、代替例外ハンドラ及び分岐命令を生成したプログラムを、コンパイル結果として出力する。

【 0 0 0 8 】

ここで、例外は、例えば、対象プログラムの実行中に、プログラム言語に予め定められた規格に反する処理が行われた場合に発生する処理である。具体的には、例外とは、対象プログラム中の命令が、配列変数を、当該配列変数に定義された範囲外の添え字を用いてアクセスを試みた場合に発生する。これに代えて、プログラム言語に予め定められた規格に反するか否かに関わらず、例外を自律的に発生させる命令によって発生してもよい。一例としては、例外は、J a v a（登録商標）言語におけるエクセプション（E x c e p t i o n）である。

【 0 0 0 9 】

なお、コンパイル対象の対象プログラムは、最適化を効率良く行うべくソースプログラムから生成された中間表現であり、例えば、J a v a（登録商標）言語のバイトコードである。これに代えて、対象プログラムは、R T L（R e g i s t e r T r a n s f e r L a n g u a g e）であってもよいし、四つ組表現であってもよい。また、対象プログラムとは、利用者により実行されるプログラム全体であってもよいし、対象プログラムのうちの一つの機能を示すモジュールであってもよい。モジュールとは、例えば、メソッド（m e t h o d）、関数（f u n c t i o n）、又は手続（p r o c e d u r e）である。

【 0 0 1 0 】

図 2 は、コンパイラ装置 1 0 のフローチャートを示す。例外ハンドラ検出部 1 0 0 は、コンパイル対象の対象プログラムを取得すると、当該対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する（S 2 0 0）。

【 0 0 1 1 】

そして、例外選択部 1 1 0 は、例外ハンドラ検出部 1 0 0 により検出された複数捕捉例外ハンドラが捕捉する複数の例外のうち、複数捕捉例外ハンドラが再び

例外が発生させることにより共通の処理に移行する例外の組を選択する（S210）。好ましくは、例外選択部110は、複数捕捉例外ハンドラにおいて発生する頻度が予め定められた基準頻度以上であることを更に条件として、例外の組を選択する。例えば、例外選択部110は、複数捕捉例外ハンドラにおいて例外が発生する頻度として、複数捕捉例外ハンドラが実行される回数当たりの、複数捕捉例外ハンドラにおいて例外の組の何れかが発生する回数を検出し、検出した当該回数に基づいて、例外の組を検出してもよい。

【0012】

例外が発生する頻度を検出する方法の一例としては、コンパイラ装置10は、他の方法により一旦コンパイルされた対象プログラムが実際に実行された場合の情報に基づいて検出してもよいし、対象プログラムの制御フロー及びデータフローの情報に基づいて、例外が発生する頻度の見積もり値を算出してもよい。

【0013】

例外ハンドラ生成部120は、例外選択部110により選択された例外の組を複数捕捉例外ハンドラに代わって捕捉して共通の処理に移行する代替例外ハンドラを生成する（S220）。更に、例外ハンドラ生成部120は、生成した当該代替例外ハンドラにおいて、共通の処理に移行させる分岐命令を生成し、生成した分岐命令によって、共通の処理に移行させる（S230）。また、例外ハンドラ生成部120は、対象プログラムの実行結果を保持するべく、複数捕捉例外ハンドラにおいて実行される処理と略同一の処理を、代替例外ハンドラに複写する。

【0014】

このように、コンパイラ装置10は、複数捕捉例外ハンドラに代わって例外を捕捉し、共通の処理に移行させる代替例外ハンドラを生成し、当該代替例外ハンドラにおいて、共通の処理に移行させる分岐命令を生成する。これにより、コンパイラ装置10は、捕捉した例外を再び発生させる処理を省略し、単なる分岐命令により共通の処理に移行させることができる。

【0015】

また、本図で示した一連の処理が行われるタイミングは、コンパイル対象の対

象プログラムの実行が開始される前に限定されない。例えば、コンパイラ装置 1 0 は、一旦コンパイルされた対象プログラムの実行中に、代替例外ハンドラを生成してもよい。より具体的には、例外選択部 1 1 0 は、対象プログラムの実行中に、複数捕捉例外ハンドラにおいて発生する頻度が予め定められた基準頻度以上である例外の組を選択する。そして、例外選択部 1 1 0 は、選択した当該例外の組のうち、複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する。これにより、例外ハンドラ生成部 1 2 0 は、対象プログラムの実行中に必要に応じて代替例外ハンドラを生成することができる。即ち、コンパイラ装置 1 0 とは、コンパイル対象の対象プログラムの実行が開始される前に当該対象プログラムを最適化する機能のみならず、当該対象プログラムの実行中に必要に応じて対象プログラムを最適化する機能、例えば、ランタイムライブラリとして実現される機能をも含む。

【 0 0 1 6 】

図 3 は、コンパイル対象の対象プログラムの一例を示す。1 行目から 1 3 行目に示した括弧は、1 3 行目に示した例外ハンドラにより例外が捕捉される例外捕捉範囲を示している。2 行目から 9 行目に示した括弧は、9 行目から 1 2 行目に示した例外ハンドラによる例外捕捉範囲を示している。即ち、本図の対象プログラムにおいて、2 行目から 9 行目の例外捕捉範囲において発生した例外は、9 行目から 1 2 行目に示した例外ハンドラにより捕捉される。更に、9 行目から 1 2 行目に示した例外ハンドラにおいて発生した例外は、1 3 行目の例外ハンドラにより捕捉される。ただし、1 3 行目の例外ハンドラは、9 行目から 1 2 行目に示した例外ハンドラにおいて発生した例外のうち、例外 1 及び例外 2 を捕捉する。

【 0 0 1 7 】

また、3 行目及び 4 行目の命令は、例外 1 を発生させる命令である。5 行目及び 6 行目の命令は、例外 2 を発生させる命令である。7 行目及び 8 行目の命令は、例外 3 を発生させる命令である。また、1 0 行目及び 1 1 行目の命令は、9 行目から 1 2 行目に示した例外ハンドラにより捕捉された例外を、再び発生させる命令である。本図で示すように、例外を自律的に発生させる命令は、例外の種類を指定する。例えば、3 行目及び 4 行目の命令は、例外 1 という種類の例外を発

生させる。そして、例外ハンドラは、指定した例外の種類に該当する例外が発生した場合に、処理を開始する。例えば、13行目の例外ハンドラは、例外検出範囲において発生した例外のうち、指定した例外に該当する例外1及び例外2を捕捉し、13行目の処理を開始する。

【0018】

例外ハンドラ検出部100は、本図に示した対象プログラムを取得すると、複数捕捉例外ハンドラとして、9行目から12行目に示した例外ハンドラを検出する。そして、例外選択部110は、複数捕捉例外ハンドラが捕捉する複数の例外、即ち、例外1、例外2、及び例外3のうち、複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する。例えば、複数捕捉例外ハンドラが、例外1及び例外2の何れを発生させた場合であっても、13行目に示した共通の処理に移行するので、例外選択部110は、共通の処理に移行する例外の組として、例外1及び例外2を選択する。

【0019】

なお、本図の例において、複数捕捉例外ハンドラとは、J a v a（登録商標）言語等において、例外検出範囲において発生した何れの例外をも捕捉する `finally` 節である。これに代えて、複数捕捉例外ハンドラは、J a v a（登録商標）言語において、複数の例外を捕捉する `catch` 節であってもよい。例えば、J a v a（登録商標）言語において、例外の種類は、例外の種類を識別するオブジェクトにより表される。また、`catch` 節は、指定された例外のオブジェクトの子クラスとなる全てのオブジェクトを捕捉する。即ち、複数捕捉例外ハンドラは、複数のオブジェクトの親クラスであるオブジェクトとして表される例外を捕捉する `catch` 節であってもよい。

【0020】

また、例外ハンドラが例外を捕捉するとは、当該例外が発生したことを条件に、当該例外ハンドラの処理が開始される旨を示す。即ち、9行目から12行目の例外ハンドラが例外1を捕捉するとは、例外1が発生したことを条件に9行目から12行目の例外ハンドラの処理が開始される旨を示す。

【0021】

図4は、コンパイラ装置10による最適化後の対象プログラムの一例を示す。例外ハンドラ生成部120は、例外選択部110により選択された組である例外1及び例外2を捕捉して共通の処理に移行する代替例外ハンドラ、例えば、9行目及び10行目の命令を生成する。そして、例外ハンドラ生成部120は、複数捕捉例外ハンドラに代えて、代替例外ハンドラにおいて、例外選択部110により選択された組の例外である例外1及び例外2を捕捉させるべく、図3の9行目から12行目に示した複数捕捉例外ハンドラに代えて、例外3のみを捕捉する例外ハンドラを11行目から12行目に生成する。そして、例外ハンドラ生成部120は、代替例外ハンドラにおいて、共通の処理に移行させる分岐命令を10行目に生成し、生成した分岐命令によって、共通の処理に移行させる。

【0022】

このように、コンパイラ装置10は、複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択し、選択した例外の組を複数捕捉例外ハンドラに代わって捕捉し、共通の処理に移行させる代替例外ハンドラを生成する。更に、コンパイラ装置10は、代替例外ハンドラにおいて、共通の処理に移行させる分岐命令を生成する。これにより、コンパイラ装置10は、例外の発生により必要となる処理を省略し、単なる分岐命令により処理を移行させることができる。

【0023】

図5(a)は、コンパイル対象の対象プログラムの他の例である。コンパイラ装置10は、関数500-1~Nを対象プログラムとして取得する。関数500-1は、例外ハンドラ510-1と、例外ハンドラ510-1の例外検出範囲において関数500-2を呼び出すコール(call)命令とを含む。関数500-2は、例外ハンドラ510-2と、例外ハンドラ510-2の例外検出範囲において関数500-3を呼び出すコール(call)命令を含む。関数500-3からN-1も関数500-1と同様であり、関数500-3からN-1は、この順に順次呼び出されることにより実行を開始する。また、関数500-Nは、関数500-(N-1)から呼び出されて実行を開始し、例外ハンドラ510-Nと、例外ハンドラ510-Nの例外検出範囲において例外を発生させる例外発

生命令520を含む。なお、関数とは、対象プログラムの作成者により予め定められた対象プログラムの部分であり、メソッド (method) 又は手続 (procedure) であってもよい。

【0024】

例外ハンドラ検出部100は、この対象プログラムを取得すると、一の複数捕捉例外ハンドラである例外ハンドラ510-Nと、例外ハンドラ510-Nにおいて発生した少なくとも一の例外を捕捉する他の複数捕捉例外ハンドラである例外ハンドラ510-2との2つの複数捕捉例外ハンドラを検出する。そして、例外選択部110は、例外ハンドラ510-Nが捕捉する複数の例外のうち、例外ハンドラ510-Nが再び例外を発生させることにより例外ハンドラ510-2の処理に移行し、かつ例外ハンドラ510-2において捕捉した例外を再び発生させることにより共通の処理に移行する例外の組を選択する。例えば、本図に例において、例外1が発生した場合には、例外ハンドラ510-2により捕捉され、かつ例外ハンドラ510-2から例外ハンドラ510-1に処理が移行するので、例外選択部110は、例外1を選択する。

【0025】

これを受けて、例外ハンドラ生成部120は、この2つの複数捕捉例外ハンドラのそれぞれに対応付けて、2つの代替例外ハンドラのそれぞれを生成し、この2つの複数捕捉例外ハンドラのそれぞれに代えて、対応する2つの代替例外ハンドラのそれぞれにおいて、例外選択部110により選択された例外1を捕捉させる。

【0026】

図5(b)は、コンパイル対象の対象プログラムにおける各命令が、発生又は捕捉する例外の種類を、関数呼び出しおよび例外捕捉範囲の入れ子の深さに対応付けて示す。例えば、例外発生命令520は、例外1を発生させる命令であり、例外ハンドラ510-2及び例外ハンドラ510-Nのそれぞれは、例外1、例外2、及び例外3を捕捉する命令である。また、例外ハンドラ510-1は、例外1を捕捉する命令である。

【0027】

また、例外ハンドラ 5 1 0 - 1 は、関数呼び出しにより呼び出された関数 5 0 0 - 2 の処理中に発生した例外 1 を、関数呼び出しの外で捕捉する。また、例外ハンドラ 5 1 0 - 2 は、例外ハンドラ 5 1 0 - 1 を含む関数から呼び出された関数の内部で発生した例外を捕捉する。即ち、例外ハンドラ 5 1 0 - 2 は、関数 5 0 0 - 2 から関数 5 0 0 - N で発生した例外を捕捉する。また、例外ハンドラ 5 1 0 - N は、例外発生命令 5 2 0 が発生させる例外を捕捉する。

即ち、例外ハンドラ 5 1 0 - 2 は、関数呼び出しの入れ子が例外ハンドラ 5 1 0 - 1 より深い。更に、例外ハンドラ 5 1 0 - 2 は、例外検出範囲の入れ子が例外ハンドラ 5 1 0 - 1 より深い。同様に、例外ハンドラ 5 1 0 - 3 から N は、この順に、関数呼び出し及び例外検出範囲の入れ子が深くなる。

【 0 0 2 8 】

このように、複数捕捉例外ハンドラの一例である例外ハンドラ 5 1 0 - N は、共通の処理の一例である例外ハンドラ 5 1 0 - 1 に処理を移す過程で、他の例外ハンドラを経由する。従って、例外ハンドラ 5 1 0 - N から例外ハンドラ 5 1 0 - 1 に移行する間に処理が経由する他の例外ハンドラの数が多い場合に、例外が発生してから共通の処理に処理が移るまでにより多くの時間が必要となる。

また、例外ハンドラ 5 1 0 - N は、例外ハンドラ 5 1 0 - 1 に処理を移す過程で、関数 5 0 0 - N から関数 5 0 0 - 2 の関数呼び出しから復帰する処理を経る。従って、例外ハンドラ 5 1 0 - 1 から例外ハンドラ 5 1 0 - N に至る関数呼び出しの入れ子の深さがより深い場合に、例外が発生してから共通の処理に処理が移るまでにより多くの時間が必要となる。

【 0 0 2 9 】

上記の処理に要する時間を減少させるべく、好ましくは、例外選択部 1 1 0 は、例外ハンドラ 5 1 0 - N から例外ハンドラ 5 1 0 - 1 に移行する間に処理が経由する他の例外ハンドラの数、予め定められた数より大きいことと、例外ハンドラ 5 1 0 - 1 から例外ハンドラ 5 1 0 - N に至る関数呼び出しの入れ子の深さが予め定められた数より大きいことを更に条件として、例外の組を選択する。

即ち、例外選択部 1 1 0 は、例外ハンドラ 5 1 0 - N が捕捉する複数の例外のうち、例外ハンドラ 5 1 0 - N が再び例外を発生させることにより例外ハンドラ

510-1に移行し、例外ハンドラ510-Nから例外ハンドラ510-1に移行する間に処理が経由する他の例外ハンドラの数、予め定められた数より大きく、かつ例外ハンドラ510-1から例外ハンドラ510-Nに至る関数呼び出しの入れ子の深さが予め定められた数より大きい、例外の組合せを選択する。

なお、複数捕捉例外ハンドラが3つ以上検出された場合も同様であり、例えば、例外選択部110は、第1の複数捕捉例外ハンドラから、第2の複数捕捉例外ハンドラを経由して、第3の複数捕捉例外ハンドラに移行する間に処理が経由する他の例外ハンドラの数、予め定められた数より大きいことと、第1の複数捕捉例外ハンドラから、第2の複数捕捉例外ハンドラを経由して、第3の複数捕捉例外ハンドラに至る関数呼び出しの入れ子の深さが予め定められた数より大きいことを更に条件として、例外の組を選択する。

【0030】

図6は、コンパイラ装置10のハードウェア構成の一例を示す。実施形態に係るコンパイラ装置10は、ホストコントローラ1082により相互に接続されるCPU1000、RAM1020、グラフィックコントローラ1075、及び表示装置1080を有するCPU周辺部と、入出力コントローラ1084によりホストコントローラ1082に接続される通信インターフェイス1030、ハードディスクドライブ1040、及びCD-ROMドライブ1060を有する入出力部と、入出力コントローラ1084に接続されるROM1010、フレキシブルディスクドライブ1050、及び入出力チップ1070を有するレガシー入出力部とを備える。

【0031】

ホストコントローラ1082は、RAM1020と、高い転送レートでRAM1020をアクセスするCPU1000及びグラフィックコントローラ1075とを接続する。CPU1000は、ROM1010及びRAM1020に格納されたコンパイラプログラムに基づいて動作し、各部の制御を行う。グラフィックコントローラ1075は、CPU1000等がRAM1020内に設けたフレームバッファ上に生成する画像データを取得し、表示装置1080上に表示させる。これに代えて、グラフィックコントローラ1075は、CPU1000等が生

成する画像データを格納するフレームバッファを、内部に含んでもよい。

【0032】

入出力コントローラ1084は、ホストコントローラ1082と、比較的高速な入出力装置である通信インターフェイス1030、ハードディスクドライブ1040、及びCD-ROMドライブ1060を接続する。通信インターフェイス1030は、ネットワークを介して他の装置と通信する。ハードディスクドライブ1040は、コンパイラ装置10が使用するコンパイラプログラム及びデータを格納する。CD-ROMドライブ1060は、CD-ROM1095からコンパイラプログラム又はデータを読み取り、RAM1020を介して入出力チップ1070に提供する。

【0033】

また、入出力コントローラ1084には、ROM1010と、フレキシブルディスクドライブ1050や入出力チップ1070等の比較的低速な入出力装置とが接続される。ROM1010は、コンパイラ装置10の起動時にCPU1000が実行するブートプログラムや、コンパイラ装置10のハードウェアに依存するプログラム等を格納する。フレキシブルディスクドライブ1050は、フレキシブルディスク1090からコンパイラプログラム又はデータを読み取り、RAM1020を介して入出力チップ1070に提供する。入出力チップ1070は、フレキシブルディスク1090や、例えばパラレルポート、シリアルポート、キーボードポート、マウスポート等を介して各種の入出力装置を接続する。

【0034】

コンパイラ装置10に提供されるコンパイラプログラムは、フレキシブルディスク1090、CD-ROM1095、又はICカード等の記録媒体に格納されて利用者によって提供される。コンパイラプログラムは、記録媒体から読み出され、入出力チップ1070を介してコンパイラ装置10にインストールされ、コンパイラ装置10において実行される。

【0035】

コンパイラ装置10にインストールされて実行されるコンパイラプログラムは、例外ハンドラ検出モジュールと、例外選択モジュールと、例外ハンドラ生成モ

ジュールとを含む。各モジュールがコンパイラ装置 1 0 に働きかけて行わせる動作は、図 1 から図 5 において説明したコンパイラ装置 1 0 における、対応する部材の動作と同一であるから、説明を省略する。

【 0 0 3 6 】

以上を示したプログラム又はモジュールは、外部の記憶媒体に格納されてもよい。記憶媒体としては、フレキシブルディスク 1 0 9 0、CD-ROM 1 0 9 5 の他に、DVD や PD 等の光学記録媒体、MD 等の光磁気記録媒体、テープ媒体、IC カード等の半導体メモリ等を用いることができる。また、専用通信ネットワークやインターネットに接続されたサーバシステムに設けたハードディスク又は RAM 等の記憶装置を記録媒体として使用し、ネットワークを介してコンパイラプログラムをコンパイラ装置 1 0 に提供してもよい。

【 0 0 3 7 】

このように、コンパイラ装置 1 0 は、複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択し、選択した例外の組を複数捕捉例外ハンドラに代わって捕捉し、共通の処理に移行させる代替例外ハンドラを生成する。更に、コンパイラ装置 1 0 は、代替例外ハンドラにおいて、共通の処理に移行させる分岐命令を生成する。これにより、コンパイラ装置 1 0 は、例外の発生により必要となる処理を省略し、単なる分岐命令により処理を移行させることができる。更に、コンパイラ装置 1 0 は、部分冗長性除去等の他の最適化処理が最適化の対象とする対象プログラム中の領域であるコンパイルスコープを広げることができるので、最適化の効率を高めることができる。

【 0 0 3 8 】

以上、本発明を実施形態を用いて説明したが、本発明の技術的範囲は上記実施形態に記載の範囲には限定されない。上記実施形態に、多様な変更または改良を加えることができる。そのような変更または改良を加えた形態も本発明の技術的範囲に含まれ得ることが、特許請求の範囲の記載から明らかである。

【 0 0 3 9 】

以上を示した実施形態によると、以下の各項目に示すコンパイラ装置、コンパイラプログラム、記録媒体、及びコンパイル方法を実現できる。

【0040】

(項目1) コンパイル対象のプログラムである対象プログラムの例外処理を最適化するコンパイラ装置であって、前記対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハンドラ検出部と、検出された前記複数捕捉例外ハンドラが捕捉する前記複数の例外のうち、前記複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する例外選択部と、前記例外選択部により選択された例外の組を前記複数捕捉例外ハンドラに代わって捕捉し、前記共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成部とを備えるコンパイラ装置。

(項目2) 前記例外ハンドラ生成部は、前記代替例外ハンドラにおいて、前記共通の処理に移行させる分岐命令を生成し、生成した前記分岐命令によって前記共通の処理に移行させる項目1記載のコンパイラ装置。

【0041】

(項目3) 前記例外選択部は、前記複数捕捉例外ハンドラにおいて発生する頻度が予め定められた基準頻度以上であり、かつ前記共通の処理に移行する例外の組を選択する項目1記載のコンパイラ装置。

(項目4) 前記例外選択部は、前記複数捕捉例外ハンドラにおいて例外が発生する頻度として、前記複数捕捉例外ハンドラが実行される回数当たりの、前記複数捕捉例外ハンドラにおいて前記例外の組の何れかが発生する回数を検出する項目3記載のコンパイラ装置。

(項目5) 前記共通の処理は、関数呼び出しにより呼び出された関数の処理中に発生した例外を前記関数呼び出しの外で捕捉する処理を含み、前記複数例外捕捉ハンドラは、前記関数呼び出しの内部で例外を捕捉し、前記例外選択部は、前記共通の処理から前記複数捕捉例外ハンドラに至る関数呼び出しの入れ子の深さが、予め定められた数より大きいことを更に条件として、前記例外の組を選択する項目1記載のコンパイラ装置。

【0042】

(項目6) 前記例外選択部は、前記複数捕捉例外ハンドラから前記共通の処理

に移行する間に処理が経由する他の例外ハンドラの数、予め定められた数より大きいことを更に条件として、前記例外の組を選択する項目 1 記載のコンパイラ装置。

(項目 7) 前記共通の処理は、関数呼び出しにより呼び出された関数の処理中に発生した例外を前記関数呼び出しの外で捕捉する処理を含み、前記複数例外捕捉ハンドラは、前記関数呼び出しの内部で例外を捕捉し、前記例外選択部は、前記共通の処理から前記複数捕捉例外ハンドラに至る関数呼び出しの入れ子の深さと、前記複数捕捉例外ハンドラから前記共通の処理に移行する間に処理が経由する他の例外ハンドラの数とに更に基づき、前記例外の組を選択する項目 1 記載のコンパイラ装置。

(項目 8) 前記例外ハンドラ検出部は、一の前記複数捕捉例外ハンドラと、前記一の複数捕捉例外ハンドラにおいて発生した少なくとも一の例外を捕捉する他の前記複数捕捉例外ハンドラとの 2 つの前記複数捕捉例外ハンドラを検出し、前記例外選択部は、前記一の複数捕捉例外ハンドラが捕捉する複数の例外のうち、前記他の複数捕捉例外ハンドラにおいて捕捉した例外を再び発生させることにより前記共通の処理に移行する例外の組を選択し、前記例外ハンドラ生成部は、前記 2 つの複数捕捉例外ハンドラのそれぞれに対応付けて、2 つの前記代替例外ハンドラのそれぞれを生成し、前記 2 つの複数捕捉例外ハンドラのそれぞれに代えて、対応する前記 2 つの代替例外ハンドラのそれぞれにおいて、前記例外選択部により選択された例外の組を捕捉させる項目 1 記載のコンパイラ装置。

【 0 0 4 3 】

(項目 9) コンパイル対象のプログラムである対象プログラムの例外処理を最適化するコンパイラ装置としてコンピュータを機能させるコンパイラプログラムであって、前記コンピュータを、前記対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハンドラ検出部と、検出された前記複数捕捉例外ハンドラが捕捉する前記複数の例外のうち、前記複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する例外選択部と、前記例外選択部により選択された例外の組を前記複数捕

捉例外ハンドラに代わって捕捉し、前記共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成部として機能させるコンパイラプログラム。

(項目 1 0) 前記共通の処理は、関数呼び出しにより呼び出された関数の処理中に発生した例外を前記関数呼び出しの外で捕捉する処理を含み、前記複数例外捕捉ハンドラは、前記関数呼び出しの内部で例外を捕捉し、前記例外選択部は、前記共通の処理から前記複数捕捉例外ハンドラに至る関数呼び出しの入れ子の深さと、前記複数捕捉例外ハンドラから前記共通の処理に移行する間に処理が経由する他の例外ハンドラの数とに更に基づき、前記例外の組を選択する項目 9 記載のコンパイラプログラム。

【 0 0 4 4 】

(項目 1 1) 前記例外ハンドラ検出部は、一の前記複数捕捉例外ハンドラと、前記一の複数捕捉例外ハンドラにおいて発生した少なくとも一の例外を捕捉する他の前記複数捕捉例外ハンドラとの 2 つの前記複数捕捉例外ハンドラを検出し、前記例外選択部は、前記一の複数捕捉例外ハンドラが捕捉する複数の例外のうち、前記他の複数捕捉例外ハンドラにおいて捕捉した例外を再び発生させることにより前記共通の処理に移行する例外の組を選択し、前記例外ハンドラ生成部は、前記 2 つの複数捕捉例外ハンドラのそれぞれに対応付けて、2 つの前記代替例外ハンドラのそれぞれを生成し、前記 2 つの複数捕捉例外ハンドラのそれぞれに代えて、対応する前記 2 つの代替例外ハンドラのそれぞれにおいて、前記例外選択部により選択された例外の組を捕捉させる項目 9 記載のコンパイラプログラム。

(項目 1 2) 項目 9 から項目 1 1 の何れかに記載のコンパイラプログラムを記録した記録媒体。

【 0 0 4 5 】

(項目 1 3) コンパイル対象のプログラムである対象プログラムの例外処理を最適化するコンパイラ装置としてコンピュータを動作させるコンパイル方法であって、前記対象プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハンドラ検出段階と、検出された前記複数捕捉例外ハンドラが捕捉する前記複数の例外のうち、前記複数捕捉例外ハンドラが再び例外を

発生させることにより共通の処理に移行する例外の組を選択する例外選択段階と、前記例外選択部により選択された例外の組を前記複数捕捉例外ハンドラに代わって捕捉し、前記共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成段階とを備えるコンパイル方法。

【 0 0 4 6 】

【発明の効果】

上記説明から明らかなように、本発明によれば例外処理を最適化することができる。

【図面の簡単な説明】

【図 1】

図 1 は、コンパイラ装置 1 0 の機能ブロック図を示す。

【図 2】

図 2 は、コンパイラ装置 1 0 のフローチャートを示す。

【図 3】

図 3 は、コンパイル対象の対象プログラムの一例を示す。

【図 4】

図 4 は、コンパイラ装置 1 0 による最適化後の対象プログラムの一例を示す。

【図 5】

図 5 (a) は、コンパイル対象の対象プログラムの他の例である。

図 5 (b) は、コンパイル対象の対象プログラムにおける各命令が、発生又は捕捉する例外の種類を、関数呼び出しおよび例外捕捉範囲の入れ子の深さに対応付けて示す。

【図 6】

図 6 は、コンパイラ装置 1 0 のハードウェア構成の一例を示す。

【符号の説明】

1 0 コンパイラ装置

1 0 0 例外ハンドラ検出部

1 1 0 例外選択部

1 2 0 例外ハンドラ生成部

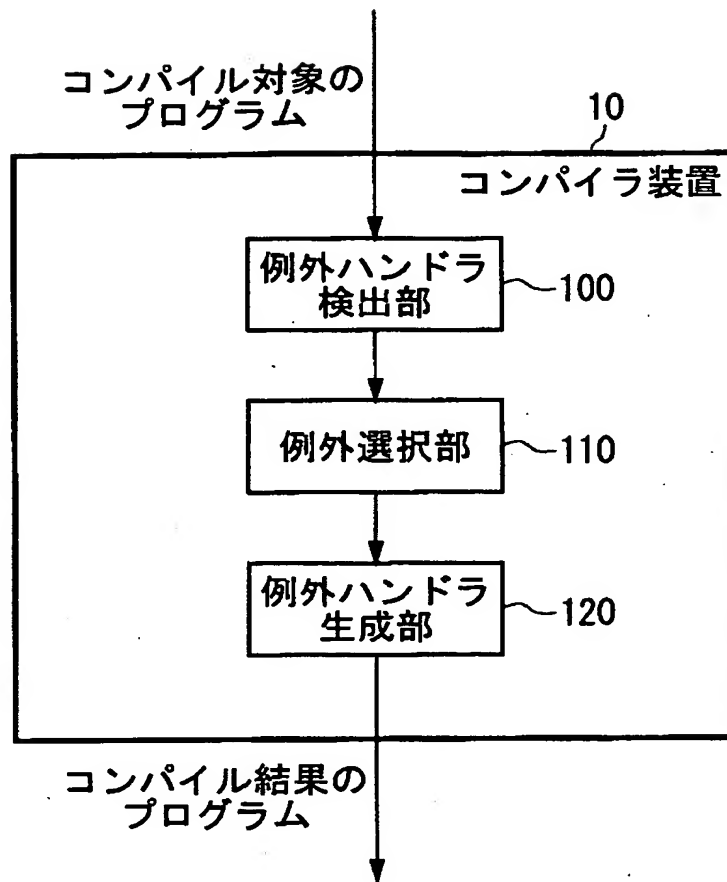
5 0 0 関数

5 1 0 例外ハンドラ

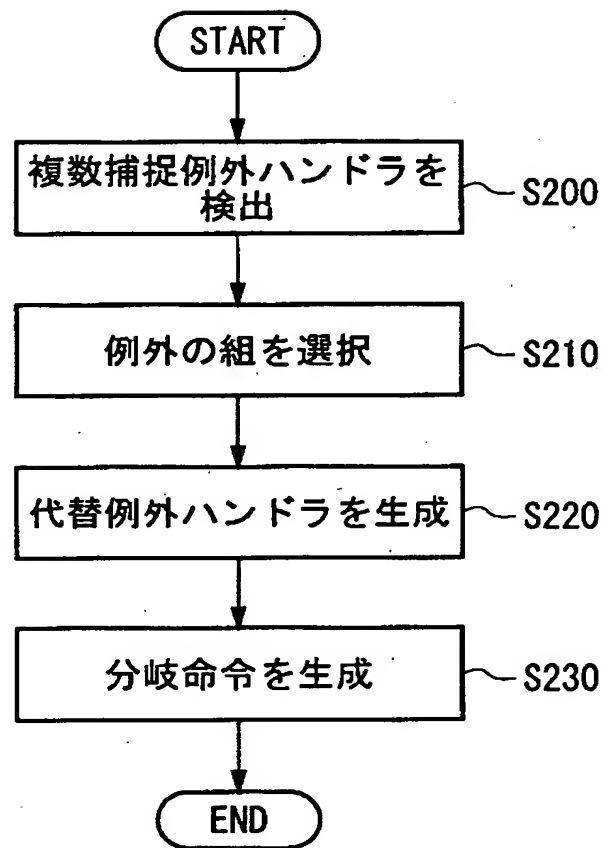
5 2 0 例外発生命令

【書類名】 図面

【図 1】



【図 2】



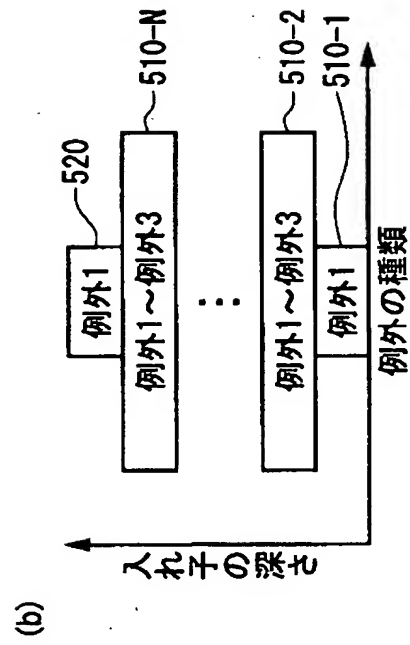
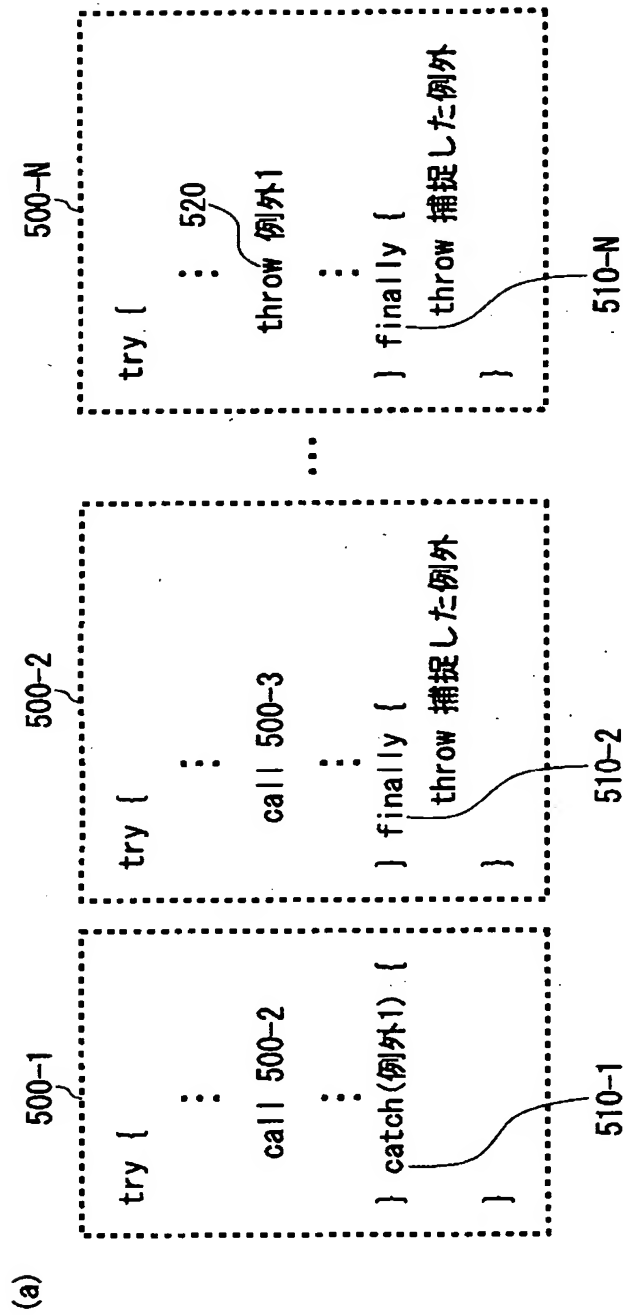
【図 3】

```
1: try {  
2:   try {  
      ⋮  
3:     L1=例外1  
4:     throw L1  
      ⋮  
5:     L2=例外2  
6:     throw L2  
      ⋮  
7:     L3=例外3  
8:     throw L3  
      ⋮  
9:   } finally {  
      ⋮  
10:    L4=捕捉した例外  
11:    throw L4  
      ⋮  
12:  }  
13:} catch (例外1, 例外2) {      }
```

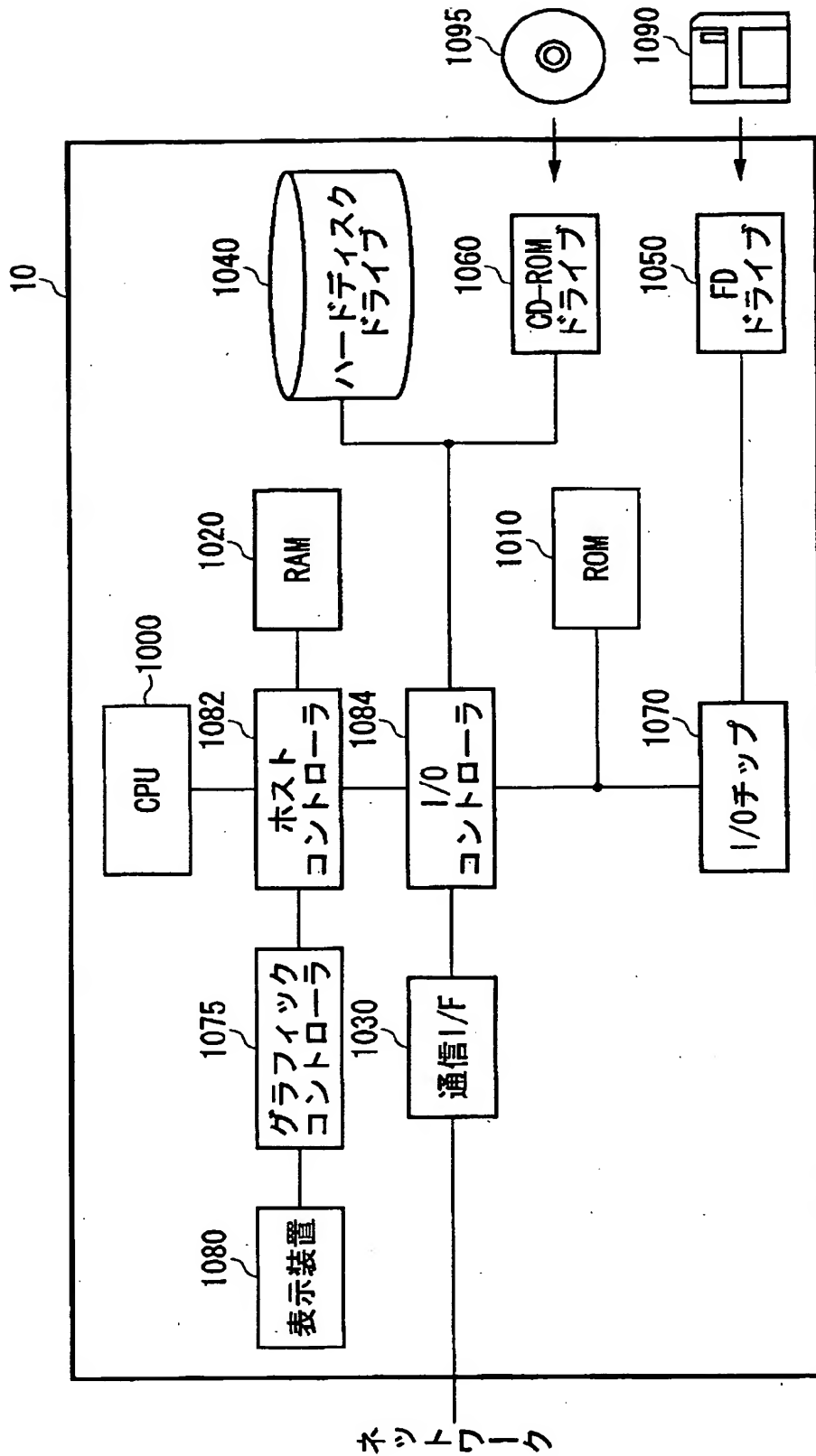
【図 4】

```
1: try {  
2:   try {  
      ⋮  
3:     L1=例外1  
4:     throw L1  
      ⋮  
5:     L2=例外2  
6:     throw L2  
      ⋮  
7:     L3=例外3  
8:     throw L3  
      ⋮  
9:   } catch (例外1, 例外2) {  
      ⋮  
10:    13行目へ分岐する分岐命令  
11:  } catch (例外3) {  
      ⋮  
12:  }  
13:} catch (例外1, 例外2) {      }
```

【図 5】



【図 6】



【書類名】 要約書

【要約】

【課題】 例外処理を最適化する。

【解決手段】 プログラムの例外処理を最適化するコンパイラ装置は、プログラムで発生した例外を捕捉する例外ハンドラのうち、互いに異なる複数の例外を捕捉し、かつ捕捉した例外を再び発生させる複数捕捉例外ハンドラを検出する例外ハンドラ検出部と、検出された複数捕捉例外ハンドラが捕捉する複数の例外のうち、複数捕捉例外ハンドラが再び例外を発生させることにより共通の処理に移行する例外の組を選択する例外選択部と、例外選択部により選択された例外の組を複数捕捉例外ハンドラに代わって捕捉し、共通の処理に移行させる代替例外ハンドラを生成する例外ハンドラ生成部とを備える。

【選択図】 図 1

認定・付加情報

特許出願の番号	特願 2003-079010
受付番号	50300464496
書類名	特許願
担当官	末武 実 1912
作成日	平成15年 5月 6日

<認定情報・付加情報>

【特許出願人】

【識別番号】	390009531
【住所又は居所】	アメリカ合衆国10504、ニューヨーク州 アーモンク ニュー オーチャード ロード
【氏名又は名称】	インターナショナル・ビジネス・マシーンズ・コーポレーション

【代理人】

【識別番号】	100091568
【住所又は居所】	神奈川県大和市下鶴間1623番地14 日本アイ・ビー・エム株式会社 大和事業所内
【氏名又は名称】	市位 嘉宏

【代理人】

【識別番号】	100108501
【住所又は居所】	神奈川県大和市下鶴間1623番14 日本アイ・ビー・エム株式会社 知的所有権
【氏名又は名称】	上野 剛史

【復代理人】

【識別番号】	100104156
【住所又は居所】	東京都新宿区新宿1丁目24番12号 東信ビル 6階 龍華国際特許事務所
【氏名又は名称】	龍華 明裕

【代理人】

【識別番号】	100086243
【住所又は居所】	神奈川県大和市下鶴間1623番地14 日本アイ・ビー・エム株式会社 大和事業所内
【氏名又は名称】	坂口 博

次頁無

出 願 人 履 歴 情 報

識別番号 [390009531]

1. 変更年月日 2002年 6月 3日

[変更理由] 住所変更

住 所 アメリカ合衆国10504、ニューヨーク州 アーモンク ニ
ュー オーチャード ロード

氏 名 インターナショナル・ビジネス・マシーンズ・コーポレーショ
ン